REALM OF RACKET
Learn to Program, One Game at a Time!

Forrest Bice, Rose DeMaio, Spencer Florence, Feng-Yun Mimi Lin, Scott Lindeman, Nicole Nussbaum, Eric Peterson, Ryan Plessner
David Van Horn | Conrad Barski, MD
Matthias Felleisen
Index

Numbers & Symbols

= function, 55
#:mutable keyword, 132
#:opaque keyword, 217
#:prefab keyword, 215
#:transparent keyword, 50
#:when keyword, 157
#%app top level form, 277, 279–280
#%datum top level form, 277
#%module-begin top level form, 277–280
#%top-interaction top level form, 278
#f value, 57
#lang lazy, 198
#lang racket, 21
#lang racket/base, 266
#lang s-exp, 275
#t value, 57
/force function, 279
+force function, 279
`(quasiquote), for lists, 276
'(quote), for data, 39, 44
,(unquote), for quasiquoted lists, 276
,@(unquote-splicing), for quasiquoted lists, 276
2htdp/image library, 80

2htdp/universe library, 80. See also
big-bang; universe

A
above function, 142
above/align function, 143
accessor, 48
add function, 186
add1 function, 33
add-3-to-state function, 81
add-board-to-scene function, 176
add-dice-to function, 184
add-path function, 243
add-player function, 247
add-player-info function, 176
add-players function, 243
add-progress-bar function, 240
add-spectator function, 252
add-territory function, 177
add-waypoint function, 252
add-waypoint* function, 243
age-goo function, 102
AI (artificial intelligence), 168, 203–208
ALGOL, 4
all-but-last function, 101
all-dead? function, 147
all-monsters-attack-player function, 150
and form, 61–64
andmap function, 118
app structure, 237
apply function, 122
argmax function, 208
arrange function, 146
artificial intelligence (AI), 168, 203–208
attackable? function, 187
attacking function, 181

B
bake-cupcakes function, 250
Barski, Conrad, i
Bayesian mathematics, 274
begin form, 271
beside function, 122, 142, 146
big-bang, 81–87, 98, 136, 174, 238
classes, 84
  name, 224, 238
  on-key, 84
  on-mouse, 238
  on-receive, 224, 238
  on-tick, 84
  register, 224, 238
  stop-when, 84
to-draw, 84
bigger function
  for Guess My Number, 33
distributed, 223
binary search, 29
body structure, 234
body-collide? function, 255
bon-appetit function, 245
boolean=? function, 55
boolean? function, 53
Booleans, 38, 63–65. See also false value;
  true value
branch, 57
brigand structure, 130. See also monster
  structure
broadcast function, 250
broadcast-universe function, 250
build-list function, 120
bundle type, 216

C

C, 4
C++, 4
can-eat function
  for Land of Lambda, 117
  for Snake, 99
case form, 150
check-= function, 70
check-equal? function, 68
check-false function, 70
check-not-equal? function, 70
check-not-false function, 70
check-pred function, 70
check-true function, 70
Church, Alonzo, 2
circle function, 243
class form, 267
case. See big-bang; cond form
client, 214
client.rkt file, 220, 221, 224
close? function, 100
closure, 197
collects, 218
color-chooser function, 177
command line execution, 266
comments, 38
complex numbers, 39, 236. See also numbers
  composite data, 55
compute-every-1000th function, 195
cond form, 58–59
conditionals
  and, 61–63, 66
  case, 140, 150
  cond, 58–59
  if, 56–58
  or, 61–63
  unless, 63, 72
  when, 63, 150
connect clause, 217
connect function
for Guess My Number, distributed, 222–223
for Hungry Henry, 245
ccons cell, 42
ccons function, 43–44
ccons? function, 107
constant, 72
count++ function, 276
count-and-print macro, 276
create-player function, 247
create-world-of-dice-and-doom function, 174
current-target function, 149
damage-monster function, 149
data types. See Booleans; bundle type;
  iworld type; mail type; number type; package type; promise type;
  sequence type; string type;
  structures
d/dx function, 121
dead? function, 107
deal-with-guess# function, 148
define form, 29–30, 71–75
define/private form, for classes, 267
define/public form, for classes, 267
define-syntax-rule form, 270
define-values form, 156
definitions. See also functions, defining
  constants, 72–73, 86
  local bindings, 73–75
  structures, 47–48, 50, 132
definitions panel, 21
delay form, 198
delayed-args-app macro, 278
dice function, 182
Dice of Doom, 165–192. See also artificial intelligence (AI)
  end of game, 189
game trees, 171–174, 181–188
handling input, 179–181
neighbors, how to find, 184–186
rendering, 176–178
rules, 166–167
dice-world struct, 171
dir? function, 103
direct-snake function, 103
disconnect function, 246
display function, 153
displayln function, 158
distribute function, 184
distributed programming, 2, 213–214, 218
domain, 274
domain-specific language
  #%app, 277
  #%module-begin, 277–280
  creation, 274–281
domain, 203, 274
  installation, 275
    delayed-lang, 279
    silly-lang, 275
draw-a-ufo-onto-an-empty-scene function, 81
draw-dice function, 178
draw-dice-world function, 175
draw-end-of-dice-world function, 175
draw-focus function, 177
draw-guess function, 224
drawing. See big-bang; images; to-draw clause
draw-square function, 73
draw-territory function, 177
drop function, 146
drop-client function, 248
drop-player function, 252
DrRacket IDE, 19–26
eat function, 100
eat-all-the-things function, 255
eighth function, 47
empty value, 43–44
empty? function, 54
empty-bundle function, 246
empty-scene function, 81
end-game-broadcast function, 256
end-of-orc-battle? function, 137
end-turn function, 147
ENIAC, 3
entree structure, 237
eq? function, 66
equal? function, 55, 65–66
equality, 55–56, 64–66, 133
equality predicate, 55–56, 64–66
error function, 72
even? function, 61
even-row function, 187
exact-integer? function, 54
except-out form, 278
execute function, 188
expt function, 39

F
false value, 57
fault tolerant, 223
feed-em-all function, 254
Felleisen, Matthias, 8
field, 48
field selector, 48
fifth function, 47
filter function, 118
findf function, 180
Findler, Robby, 8
find-move function, 180
first function, 45–47
flail function, 148
flame-thrower wielding squirrels, 199
Flatt, Matthew, 8
foldr function, 118
for form, 153
for*/list, 159
for/and, 160
for/first, 160
for/fold, 154, 156
for/last, 160
for/list, 154
for/or, 160
force function, 198
force* function, 279
for-each function, 150
form, 58
FORTRAN, 4
fourth function, 47
fresh-goo function, 102
Friedman, Dan, 8
functional programming, 131–135, 182
functions
anonymous, 113–115
calling, 21, 31–32
defining, 30–33
helper, 102, 143, 239
higher-order, 112–113, 121
recursive, 59–60, 99, 101

G
game structure
for AI, 205
for Dice of Doom, 171
game tree, 168–170. See also minimax
branches, 169
depth limit, 206
pruning, 207
games
Dice of Doom, 165–192
AI, 206–212
Guess My Number, 27–34
distributed, 219–226
GUI, 85–90
Hungry Henry, 232–261
Orc Battle, 128–152
Snake, 95–110
game-tree function
for AI, 204
for Dice of Doom, 183
gensym function, 247
g-get-dice-img function, 178
g-get-iws function, 251
get-posns-from-goo function
  for Land of Lambda, 113
  for Snake, 106
get-random-numbers function, 190
get-row function, 182
get-score function, 256
get-x function, 182
get-y function, 182
give-monster-turn-if-attack#=0
  function, 150
goo structure, 97
goo-list+scene function
  for Land of Lambda, 113
  for Snake, 106
go-to function, 252
g graphical user interface (GUI), 79–89. See also big-bang
grow function, 100
guess function
  for Guess My Number, 31
distributed, 223
  GUI, 88
Guess My Number, 27–34
distributed, 219–226
  GUI, 85–90
GUI (graphical user interface), 79–89. See also big-bang

H
handle-appetizer-message
  function, 240
handle-entree-message function, 244
handle-goto-message function, 246
handle-keys function, 224
handle-msg function
  for client, 224
  for server, 223
handler function. See big-bang; universe
handle-server-messages function, 239
heal function, 147
hexagon function, 177
higher-order functions, 112
higher-order programming. See functions;
  higher-order functions; loops;
  memoization
Hudak, Paul, 8
Hungry Henry, 231–261
  appetizer, 237, 239–241
  entree, 237, 242–245
  join, 237, 247–252
  play, 237, 251–257
hydra structure, 130. See also monster structure

I
IDE (interactive development environment), 19. See also DrRacket IDE
if form, 56–57
image? function, 53
images. See above function; above/
  align function; beside function;
  circle function; empty-scene
  function
img+scene function, 106
img-list+scene function, 106, 118
Indiana University, 8
inherit-field form, for classes, 268
init-field form, for classes, 267
initialize-monsters function, 140
initialize-orc-world function, 137
initialize-player function, 140
in-list function, 160
in-naturals function, 195
in-range function, 159
inside-of-rectangle? function, 73
instance, 48
instructions function, 142
integer? function, 54
interactions panel, 21–26
interactive development environment
  (IDE), 19. See also DrRacket IDE
interact-count-and-print macro, 278
interact-with-board function, 175
interval structure, 86
ip structure, 238
iworld type, 216
iworld=? function, 217
iworld-name function, 233

J
join structure, 237
join-add-player function, 247
join-remove function, 248

K
keep-waiting function, 249
keep-waiting? function, 249
key=? function, 87
keyword, 29
Krishnamurthi, Shriram, 8

L
lambda, 111–123. See also closure
lambda form, 113–114, 121–122
Land of Lisp, i
launch-guess-client function, 224
launch-guess-server function, 222
lazy evaluation, 194. See also memoization
delay, 198, 204–205
force, 198, 204–206
left function, 179
length function, 148
let form, 135
lets-eat function, 238
Lisp language, i
list function, 41–47
list? function, 54
list-eater, 59
list-ref function, 144
lists. See also for form
cons, 43–44
cons cell, 42–43
empty, 43–44
first, 43, 45–47
for/list, 154
list-eaters, 59
loop functions
andmap, 116
filter, 114–115
foldl, 118
foldr, 117
map, 112–113
ormap, 115
rest, 43, 45–47
traversals. See for form; list-eater; lists;
loop functions; loops; recursion
local definitions
of constants, 73–74
of functions, 75
LOCALHOST constant, 217
loop functions
andmap, 116
filter, 114–115
foldl, 118
foldr, 117
map, 112–113
ormap, 115
loops, 153–160. See also for form
in-naturals, 195
in-range, 159
lose? function, 146

M
macro programming, 270–283
macros. See also domain-specific language
gensym, 248, 273
for languages, 275–281
mail type, 216
main functions, 33
make-bundle function, 216
make-connection function, 247
make-lazy+ function, 195
make-mail function, 216
make-object function, 267
make-package function, 216
make-rectangular function, 236
map function, 118
mark function, 180
max function, 33
member function, 64
memoization, 196
memoize function, 196
memoize.v2 function, 197
message function, 142
metaprogramming. See domain-specific language; macro programming
minimax, 206
MIT, 4
module, 71, 218–223
module-level definitions, 71–73
monster structure, 141
monster-alive? function, 147
move structure, 172
move-and-eat function, 253
move-player* function, 253
move-target function, 149
move-toward-waypoint function, 254
mutate, 131
mutator, 132
my-and function, 270
my-andmap function, 116
my-build-list function, 119
my-equal? function, 66
my-filter function, 114
my-first-program.rkt file, 23
my-foldl function, 117, 119
my-if function, 280
my-let.v1 macro, 273
my-map function, 112
my-or macro, 271
my-ormap function, 115

N
name clause, 224
named-player function, 247
neighbors function, 186
new form, 268
next-head function, 101
next-interval function, 223
next-pit function, 98
ninth function, 47

no-more-moves-in-world? function, 175
number type
complex, 39, 236
exact, 39
floating-point, 39
inexact, 40
integer, 39
natural, 74, 120, 195
rational, 39
real, 236
number? function, 53
number->string function, 88

O
object% value, 267
object-oriented language, 268
object-oriented programming classes, 267–269
classes as values, 269
methods, 266–268
objects, 267
odd? function, 61
odd-row function, 187
on-key clause, 84
on-msg clause, 217
on-new clause, 217
on-receive clause, 217
on-tick clause, 84
only-in function, 257
opposite-dir? function, 104
or form, 61–64
Orc Battle, 128–152
actions, 131–136, 147–150
ending game, 146–147
orc world
initializing, 140–141
rendering, 142–146
setting up the world, 128–131
starting game, 136–139
orc structure, 130. See also monster structure
orc-world structure, 136
ormap function, 118
overlay function, 88, 107
overlay/align function, 144
overlay/offset function, 178

P
package type, 216
Pascal, 4
pass function
  for AI, 208
  for Dice of Doom, 180
PDE (program development environment), 19
pit structure, 96
place-image function, 73
place-image/align function, 86
play structure, 237
play-add-spectator function, 252
player structure
  for Hungry Henry, 234
  for Orc Battle, 132
player-acts-on-monsters function, 138
player-dead? function, 146
player-health+ function, 134
player-remove function, 253
player-update! function, 134
posn structure, 96
posn=? function, 108
posn-move function, 101
predicate, 51–55
prefab (previously fabricated structure), 215
program development environment (PDE), 19
progress function, 256
promise type, 198
provide form, 219

Q
quasiquote (‘), for lists, 276
quote (‘), for data, 39, 44

R
Racket, i
  conventions, 30, 37, 52, 59
download, 20
etiquette, 30
history, 2–10
Racketeer, 20
racket-lang.org, 20
racket/promise library, 198
rackunit library, 68
raco, 266
random+ function, 140
random-number-of-attacks function, 140
random-quotient function, 140
random-stars function, 73
rate-moves function, 207
rate-position function, 207
rational? function, 54
reader, 37
read-eval-print loop, 21
real? function, 54
real-part function, 236
rectangle function, 143–144
recursion, 59–61
refocus-board function, 179
register clause, 217
remainder function, 195
remake-join function, 257
remove function, 100
rename-out form, 277
render function, 88
render-appetizer function, 239
render-avatar function, 243
render-entree function, 242
render-id+image function, 240
render-last-scene function, 88
render-monsters function, 144
render-orc-battle function, 137
render-orc-world function, 142
render-pit function, 105
render-player function, 143
render-scores function, 245
render-text function, 243
render-the-end function, 137
render-the-meal function, 239
renew function
  for Land of Lambda, 115
  for Snake, 102
require form, 68, 218–219
rest function, 45
restart function, 244
return-five function, 32
Rice University, 8
right function, 180
rip function, 248
roll-the-dice function, 174
rot function
  for Land of Lambda, 113
  for Snake, 102
rotate-until function, 179
rotten? function, 102
run function, 225
run.rkt file, 225

S
scope, 73
score function, 256
second function, 47
selector. See field selector
self-colliding? function, 107
semantics, 35, 38–50
sequence type, 160
serialize-universe function, 251
serve-dinner function, 257
server, 214
server.rkt file, 220, 222–223
set! form, 32
set-waypoint function, 239
seventh function, 47
S-expression type, 215
shared.rkt file, 220–221
single? function
  for Guess My Number, distributed, 223
  for Guess My Number, GUI, 88
sixth function, 47
slime structure, 130. See also monster
  structure
slither function, 100
smaller function
  for Guess My Number, 32
distributed, 223
  GUI, 87
Snake, 95–110
  ending game, 107–108
  data representation, 96–97
  functions in, 98–107, 108
snake structure, 96
snake-body function, 108
snake-change-dir function, 108
snake-head function, 108
snake-tail function, 108
snake%, 267
snake+scene function, 105
sort function, 245
squirrels, flame-thrower wielding, 199
sqrt function, 39
square function, 37
stab function, 147
start function
  for Guess My Number, 33
    GUI, 86
    for Orc Battle, 136
start-game function, 250
start-snake function, 98
state machine, 232–233
state-is-300 function, 83
status-bar function, 144
Steele, Guy, 7
stop-when clause, 84
stop-with function, 87
stream type, 160
string type, 40
string=? function, 55
string-append function, 41
string? function, 53
struct form, 47
struct inheritance, 130
struct-out form, 219
structures. See also definitions, structures creating, 47–48
field access, 48
mutation, 131–135
predicates, 52
sub1 function, 33
sum-territory function, 189
super-new form, for classes, 267
suspended computation, 194
Sussman, Gerry, 7
switch function, 183
switch-to-entree function, 241
symbol type, 39
symbol=? function, 52, 55
symbol? function, 53
syntax, 35–38. See also comments;
conditionals; define form; form
begin, 272, 276–277
let, 135, 205, 238, 272
set!, 32–34, 61, 196
syntax error, 36

T
take function, 146
teaching languages, 21
tenth function, 47
territory structure, 171
territory-build function, 181
test expression, in if expression, 57
testing, 68–70
text function, 86, 107, 142
the-ai-plays function, 208
then expression, in if expression, 57
third function, 47
thunk, 194
tick (‘), for data, 39, 44
tick-tock function, 246
time-broadcast function, 249
time? function, 241

to-draw clause, 84
ttrue value, 57
Turing, Alan, 36
type predicate, 53

U
universe, 214–223, 245
clauses, 217
on-disconnect, 245
on-msg, 222, 245
on-new, 222, 245
on-tick, 245
universe form, 214
unless form, 63
unmark function, 181
unquote ((,), for quasiquoted lists, 276
unquote-splicing ((, @), for quasiquoted lists, 276

V
values form, 155
variable bindings, 154
view, 80
void function, 132

W
wait-or-play function, 249
wall-colliding? function, 108
Wand, Mitch, 8
when form, 63
win? function, 146
winners function
for define define ’define, 74
for Dice of Doom, 189
won function, 189
world-change-dir function, 103
worlds, 80

Y
your-or macro, 272

Z
zero? function, 52
Zuse’s Z3, 3